

1 CLEMENT SETH ROBERTS (SBN 209203)
croberts@orrick.com
2 RAGHAV KRISHNAPRIYAN (SBN 273411)
rkrishnapriyan@orrick.com
3 ORRICK, HERRINGTON & SUTCLIFFE LLP
The Orrick Building
4 405 Howard Street
San Francisco, CA 94105-2669
5 Telephone: +1 415 773 5700
Facsimile: +1 415 773 5759

6 GERALD E. PORTER (SBN 344396)
gporter@orrick.com
7 ORRICK, HERRINGTON & SUTCLIFFE LLP
120 Broadway, 4th Floor
8 Santa Monica, CA 90401
9 Telephone: +1 310 633 2800

10 *Attorneys for Plaintiff* GONG.IO, INC.

11 UNITED STATES DISTRICT COURT
12 NORTHERN DISTRICT OF CALIFORNIA
13

14 GONG.IO, INC.,
15 Plaintiff,
16 v.
17 HYPERDOC INC. d/b/a RECALL.AI,
18 Defendant.
19

Case No. 5:25-cv-01026-NW

FIRST AMENDED COMPLAINT

Judge: Hon. Noël Wise
DEMAND FOR JURY TRIAL

20
21 Plaintiff Gong.io, Inc. (“Plaintiff” or “Gong”) brings this complaint to address a knowing
22 and willful campaign of patent infringement by Defendant Hyperdoc Inc., which does business as
23 Recall.ai (“Defendant” or “Recall.ai”). As alleged below, Recall.ai markets and sells functionality
24 to allow bots to join online video conferences and record them – functionality that infringes Gong’s
25 own patented inventions in this area. Moreover, Recall.ai instructs its users on exactly how to use
26 this functionality, encouraging them to infringe. And when Gong pointed out Recall.ai’s
27 infringement, Recall.ai did not even make an attempt prior to the filing of this lawsuit to articulate
28

1 a theory that it did not infringe, effectively conceding that it does what Gong’s patent claims. Gong
2 brings this lawsuit to put a stop to Recall.ai’s ongoing infringement of its intellectual property
3 rights.

4 **INTRODUCTION**

5 1. Gong.io is a leader in leveraging proprietary artificial intelligence to enable
6 businesses to capture, analyze, and act on customer interactions through a single, integrated
7 platform. Among its numerous innovations, Gong developed and deployed call recording software
8 that allows businesses to record and transcribe their online meetings. These capabilities apply
9 across a variety of web conferencing platforms, ensuring seamless functionality regardless of the
10 tools used by its customers.

11 2. Gong’s web conference recording solutions include multiple methods for capturing
12 meetings, such as a Gong bot that joins web conferences as a participant to record and document
13 the sessions. Gong’s flexible recording tools empower organizations to take full advantage of
14 videoconferencing – both internally and with their clients – secure in the knowledge that
15 information will not fall through the cracks because of the ephemerality of the medium.

16 3. Gong’s groundbreaking contributions to the field of online call recording are
17 protected by U.S. Patent No. 9,699,409 (the “’409 patent”), entitled “Recording Web Conferences.”
18 The ’409 patent, issued to Gong’s co-founders—its CEO and Chief Product Officer—confirms the
19 novelty of Gong’s innovations in the web conference recording space. The ’409 patent claims
20 priority to an application filed over eight years ago, long before web conferencing became the
21 ubiquitous business tool it is today.

22 4. Unfortunately, not all latecomers have been willing to respect Gong’s intellectual
23 property rights in this area. Recall.ai markets and sells technology whose use to record online web
24 conferences infringes one or more claims of the ’409 patent. And Recall.ai tells its customers
25 exactly how to infringe the ’409 patent, and continues to do so even after Gong explained in
26 exhaustive detail that Recall.ai’s technology is infringing.

THE PARTIES

5. Plaintiff Gong.io, Inc., is a corporation organized under the laws of Delaware, with its principal place of business located at 201 Spear St. 13th Floor, San Francisco, CA 94105. Gong is a leading innovator in artificial intelligence solutions for analyzing and optimizing business communications.

6. Defendant Hyperdoc Inc. is a corporation organized under the laws of Delaware, with its principal place of business located at 2261 Market Street #4339, San Francisco, CA 94114.

7. Recall.ai develops and markets technology products and services that directly compete with Gong’s patented offerings. For example, Recall.ai has written a tutorial explaining that “Gong’s recording bot is a signature part of their product,” and instructing users on how to “replicate Gong’s recording bot with Recall.ai.” (Amanda Zhu, *How to clone Gong’s bot recorder*, Recall.ai (Feb. 6, 2023),

<https://web.archive.org/web/20230922050851/https://www.recall.ai/post/how-to-clone-gongs-bot-recorder>.)

JURISDICTION AND VENUE

8. This action arises under the patent laws of the United States, Title 35 of the United States Code, including 35 U.S.C. § 271.

9. This Court has subject matter jurisdiction under 28 U.S.C. §§ 1331 and 1338(a) because this action arises under the patent laws of the United States.

10. This Court has personal jurisdiction over Recall.ai because Recall.ai is headquartered in California, conducts business in California, has employees and officers in California, and sells and offers for sale products and services (including the infringing technology) in California (including in this District).

11. Venue is proper in this District under the provisions of 28 U.S.C. § 1400(b) because Recall.ai has one or more regular and established places of business in this District (such as its headquarters at 2261 Market Street #4339, San Francisco, CA 94114) and has committed acts of infringement in this District.

THE PATENT-IN-SUIT

U.S. Patent No. 9,699,409

12. U.S. Patent No. 9,699,409, attached as Exhibit A, is entitled “Recording Web Conferences” and was issued by the United States Patent and Trademark Office on July 4, 2017.

13. The ’409 patent names as inventors Amit Bendov and Eilon Reshef, the co-founders of Gong. Messrs. Bendov and Reshef are Gong’s CEO and Chief Product Officer, respectively. Gong is the owner of all right, title, and interest in and to the ’409 patent.

14. The ’409 patent claims priority to U.S. Provisional Application No. 62/296,107, which was filed on February 17, 2016.

15. The ’409 patent is directed to systems and methods for recording web conferences, specifically for capturing audio, video, and screen-sharing content during virtual meetings. (See Ex. A, ’409 patent, col. 1, *ll.* 28-34.)

16. The ’409 patent emerged out of Gong’s innovations in the field of video conference recording.

17. As the ’409 patent explains, there is often a “need to document virtual meetings” using “a recording of audio and video data from the virtual meeting.” (See *id.*, col. 1, *ll.* 35-40.)

18. The ’409 patent describes and claims several ways to “unintrusively record[]” these meetings, “transparently to the participants.” (See *id.*, col. 1, *ll.* 44-46.)

19. Among those ways are techniques that involve, among other things, “identifying ... virtual conferences being operated by a conferencing system” (see *id.*, col. 1, *ll.* 57-60; see also *id.*, col. 16, *ll.* 1-3 & Fig. 2) and using a “virtual participant [that] emulates a human attendee, logging into a meeting supported by a conventional videoconferencing program” and “recording information streams of the human participants” (see *id.*, col. 1, *ll.* 46-48, 60-65; see also *id.*, col. 16, *ll.* 7-11).

20. For the reasons described further below, the ’409 patent describes and claims an unconventional technological solution to a technological problem—a problem that would never arise in the world before computers, where videoconferencing systems as we know them did not

1 exist. In doing so, it uses technology in an unconventional manner to achieve an improvement in
2 computer functionality.

3 21. To see why the '409 patent describes and claims an unconventional solution requires
4 contrasting Gong's inventions to what came before. Prior to Gong's inventions, recording an online
5 video meeting often required elaborate and often ineffective methods.

6 22. For instance, many users relied on screen capture programs like TechSmith
7 Camtasia, QuickTime Player, or Open Broadcaster Software (OBS) to record their screen and
8 system audio during a videoconference. These software tools captured whatever was on the user's
9 display.

10 23. These DIY solutions had numerous drawbacks. Some users described "trying to
11 record both sides of a Skype call" using this technique as a "nightmare" and explained that when
12 they tried to "record[] on the same machine doing the conference ... the developers all called in
13 sick." (See Comment, kozikowski (May 2014), *on* Audacity 3 Help Forum, *Record a Video*
14 *Conference*, Audacity (May 2014), [https://forum.audacityteam.org/t/record-a-video-](https://forum.audacityteam.org/t/record-a-video-conference/34052)
15 [conference/34052](https://forum.audacityteam.org/t/record-a-video-conference/34052) [<https://perma.cc/PEQ6-4HQU>].)

16 24. Such partial solutions also left much to be desired when it came to privacy, both for
17 the user performing the recording and the other users in the videoconference. Because the screen-
18 recording tool captured everything on the screen of the user running it, it had the potential to also
19 capture private information, such as private messages sent to that user during the videoconference
20 by other participants. And because other users were not necessarily notified that one of the
21 participants was running a screen-recording tool, these partial solutions had the potential to result
22 in participants being recorded without their consent.

23 25. Human error could also wreak havoc with these prior solutions. For example, users
24 had to remember to start the screen recorder every time. Forgetting to hit "Record" could be costly:
25 One user admitted that "I forgot to press record on Camtasia and record the webinar!!" (See The
26 Revit Kid (Oct. 9, 2013) *on* The Revit Kid Blog (Oct. 9, 2013),
27
28

1 <https://therevitkid.blogspot.com/2013/10/saubim-50-sale.html> [<https://perma.cc/AQN2-GSWV>].)

2 Reliance on human memory and discipline meant many calls inadvertently went unrecorded.

3 26. Because this partial solution required video conference participants to run an
4 additional, performance-intensive piece of software on their computers to record their screen, in
5 addition to their video conferencing software, it could also result in performance problems. Users
6 reported that screen recording could bog down their computers. One user complained that despite
7 using high-end hardware “[w]hen Im (sic) using the screen capture in OBS Studio my whole pc is
8 lagging. It goes down to 30 FPS.” (See Comment, overdev (July 10, 2016), *on* OBS Studio Support,
9 *screen capture slowing down my pc*, Open Broadcaster Software (July 10, 2016),
10 <https://obsproject.com/forum/threads/screen-capture-slowing-down-my-pc.50740/>
11 [<https://perma.cc/V392-4SEV>].) This performance hit made recording troublesome since the very
12 act of recording might degrade the call quality or the presenter’s ability to multitask.

13 27. Yet another challenge was capturing all sides of the conversation. Screen recording
14 tools struggled to record system audio and microphone input simultaneously. Users explained that
15 “[c]onferencing systems have critical audio management requirements and those designs make it
16 super hard to record unless recording, capture or monitoring is built-in.” (See kozikowski (May
17 2014), *on* Audacity 3 Help Forum, *Record a Video Conference*, Audacity (May 2014),
18 <https://forum.audacityteam.org/t/record-a-video-conference/34052/4> [[https://perma.cc/Q96N-](https://perma.cc/Q96N-SZPQ)
19 [SZPQ](https://perma.cc/Q96N-SZPQ)].) Users experiencing problems were told that the software they were using “tends to do
20 what you found; record only one side. If you do force it to record both sides, real-time echo
21 cancellation may crash.” (See *id.*)

22 28. Other ways to overcome these issues had their own drawbacks. These included
23 leaving a cellphone running on a user’s table or using its camera to take a video of the meeting
24 screen. These techniques resulted in the recording having poor-quality audio and video.

25 29. Using the built-in meeting recording feature offered by certain videoconferencing
26 platforms was also problematic. In the first place, those built-in recording tools are specific to a
27 particular system and cannot be used across systems. That was a problem because many video
28

1 conferencing systems did not have (and still do not have) a built-in solution while users often
 2 interact with multiple video conferencing systems depending on who they are talking to. (See
 3 Comment, kozikowski (May 2014), *on* Audacity 3 Help Forum, *Record a Video Conference*,
 4 Audacity, (May 2014), <https://forum.audacityteam.org/t/record-a-video-conference/34052>
 5 [<https://perma.cc/PEQ6-4HQU>].) Even when they did offer such a feature, it was often limited and
 6 not extensible. For example, where they existed, the built-in solutions for recording online
 7 videoconferences often did not (and still may not) produce video files that are in standardized file
 8 formats or that are portable to other pieces of software. And some corporate IT policies disable all
 9 but the basic videoconferencing functionality, meaning that no one who joined a meeting hosted
 10 by someone at such a company could record the meeting.

11 30. The inventions disclosed and claimed in the '409 patent elegantly solve these issues.

12 31. For example, the '409 patent discloses that “details of the conference to be recorded
 13 are obtained,” which avoids the need for a participant to remember to start recording the meeting.
 14 (See Ex. A, '409 patent, col. 6 *ll.* 22-24; Fig. 2; col. 16 *ll.* 1-4.)

15 32. The '409 patent provides several specific technical methods of obtaining those
 16 conference details. For example, the details “can be obtained from users’ calendar, customer
 17 relationship management (CRM) and/or call scheduling software when suitable access is granted.”
 18 (See *id.*, col. 6 *ll.* 31-34.)

19 33. The '409 patent also explains that conference details “can be obtained ... from the
 20 conference provider via an API”¹ and provides “an exemplary SOAP (Simple Object Access
 21 Protocol) to obtain conference details from WebEx servers” (i.e., a query that can be sent to the
 22 servers associated with the WebEx videoconferencing system to obtain conference details). (*Id.*
 23 col. 6 *ll.* 25-28.) The SOAP request provided in the '409 patent is reproduced below:

```
24      <?xml version="1.0" encoding="UTF-8"?
25      <serv:message
26      xmlns:xsi="http://www.w3.org/2001/XMLSchema-
```

26 ¹ An API is like a menu at a restaurant that allows a program to interact with another app or
 27 website. Just as a diner can order from a menu without needing to know how the food is prepared
 28 in the kitchen, an API allows one program to request that another perform a certain action without
 needing details on how that will be accomplished.

```

instance"
xmlns:serv="http://www.webex.com/schemas/2002/06/s
ervice"
xmlns:th="http://www.thymeleaf.org"
xmlns:extra="http://www.honeyfy.com/extra">
  <!--/*@thymesVar          id="startFrom"
  type="java.lang.String"*/-->
  <!--/*@thymesVar          id="maxItems"
  type="java.lang.Integer"*/-->
  <!--/*@thymesVar          id="startDateTime"
  type="java.lang.String"*/-->
  <header
  th:include="t1/webexintegration/header.xml::h
  eader"></header>
  <body>
    <bodyContent
    type="java:com.webex.service.binding.mee
    ting          LstsummaryMeeting"
    extra:attrnamespace="type=xsi">
      <listControl
      th:if="{maxItems!=null}">
        <startFrom
        th:if="{startFrom}"
        th:text="{startFrom}">1</star
        tFrom>
        <maximumNum
        th:text="{maxItems}">50<maxim
        umNum>
      </listControl>
      <order>
        <orderBy>STARTTIME</orderBy>
        <orderAD>ASC</orderAD>
      </order>
      <dateScope>
        <startDateStart
        th:text="{startDateTime}"></s
        tartDateStart>
      </dateScope>
    </bodyContent>
  </body>
</serv:message>

```

(See *id.* col. 12, l. 47-col. 13, l. 16.)

34. This aspect of the invention of the '409 patent solves the problem of participants failing to remember to start the video recording, by integrating recording into the participants' workflow. For example, the bot can "connect directly to your team's online calendar" and "automagically join[] and record[] all sales calls." (See Jonathan Costet, *Sales Call Recording*: 5

1 *Ways Uber-Successful Teams Use Game Tape*, Gong Blog (Apr. 23, 2021),
2 <https://www.gong.io/blog/sales-call-recording/> [<https://perma.cc/4SWA-QPCE>].)

3 35. Because the conference information is obtained from a source where it would have
4 been stored in the ordinary course, a participant does not need to remember to take extra steps to
5 start the recording.

6 36. Similarly, the '409 patent discloses recording videoconferences by using a virtual
7 entity that functions like a human participant – in other words, a bot – to join the conference and
8 automatically record the proceedings. (See Ex. A, '409 patent, Figs. 1-2; col. 1, *ll.* 44-56; col. 6, *ll.*
9 47-50; col. 8, *ll.* 25-28.) More specifically, the '409 patent discloses and claims a virtual participant
10 that is registered with the conferencing system as a co-participant in the virtual conference by
11 emulating human interactions with a graphical user interface, and thereby requires that the virtual
12 participant be located on a different computer relative to the conferencing system.

13 37. This aspect of the '409 patent solves many of the privacy, performance, and
14 technical issues that plagued preexisting attempts at recording videoconferences. For example, as
15 a result of this architecture the '409 patent allows the recording system of the invention to be used
16 with multiple videoconferencing systems with a consistent user experience, instead of being
17 restricted to those videoconferencing systems that provided in-built recording capabilities, or
18 requiring the user to adapt to the differences in recording capabilities between different
19 videoconferencing systems.

20 38. In addition, the privacy concerns associated with the recording picking up unwanted
21 notifications on a participant's screen are solved because the virtual participant sees only what any
22 human who joined the videoconference would see. It does not see private information that appears
23 only on one participant's screen, like direct messages.

24 39. The potential privacy issues associated with participants not knowing that they are
25 being recorded are also addressed by this aspect of the '409 patent. Because the virtual participant's
26 presence is visible, just like that of any human participant's, all participants know that the meeting
27
28

is being recorded. For example, in the exemplary screenshot below, the virtual participant appears as the “Hack the Box Notetaker”:



40. The performance issues are also solved because the heavy lifting of recording can be offloaded from the user’s computer. The virtual participant can run on a third-party computer or even the cloud, rather than on a user’s local machine. And because the virtual participant does not have to make use of a human participant’s microphone and speaker feeds, the problem of recording both sides of a conversation is also solved.

41. By recording meetings using a virtual participant, the recording system can capture whatever a human participant sees, and make use of that information in whatever way the maker of the recording system desires. In other words, one is not limited by the functionality that is built into the video conferencing system (e.g., Zoom, Teams, Google Meet, etc.) itself.

42. In addition to claiming these technological measures, the specification elaborates on how these technological measures can be implemented.

43. For example, the ’409 patent includes seven “Computer Program Listings,” that provide exemplary computer code, data, and API requests. (Ex. A, ’409 patent, col. 12 *l.* 45-col. 15, *l.* 62.)

44. For example, the Computer Program Listings include exemplary Java code that uses an API to interact with a video conferencing system.

```

private final IUIAutomationCondition
IS_AUDIO_CONNECTION=createNameCondition("Audio Connection");
private final in IUIAutomationCondition
IS_CLOSE=createNameCondition("Close");
private final in IUIAutomationCondition
IS_OK=createNameCondition("OK");
default IUIAutomationCondition
createNameCondition(String uiAutomationPropertyName){
    return
    createPropertyCondition(PropertyID.UIA_NamePropertyId,
    uiAutomationPropertyName);
}
default IUIAutomationCondition
createPropertyCondition(int propertyId, String value) {
    return
    getUIAutomation().createPropertyCondition(propertyId,
    value);
}
private boolean closeAudioConnectionPopup( ) {
    final IUIAutomationElement
    audioConnectionPopup=findFirstChild(meetingwindowElement,
    IS_AUDIO_CONNECTION);
    if (audioConnectionPopup!=null) {
        logger.warn(INTERRUPTION_MESSAGE,
        MethodNameGetter.getMethodName( ));
        return findAndClick(audioConnectionPopup,
        IS_CLOSE, "close button", 2) ||
        findAndClick(audioConnectionPopup, IS_OK, "OK
        button", 2);
    }
    return false;
}

```

(See *id.*, col. 13, ll. 20-52.)

45. As another example, the Computer Program Listings include exemplary code for automatic interaction with a videoconferencing system over the web.

```

private ChromeDriverExecutorService
startChromeBrowserWithSeleniumDriver(Path
chromeDriverLocation,
ChromeDriverExecutorService.HttpResponseFilter
httpResponseFilter, Map<String, Object>connectorParams) {
    final
    Optional<ChromeDriverExecutorService.ProxySettings>
    optionalProxySettings = isProxyEnabled
    (connectorParams) ? Optional.of(new
    ChromeDriverExecutorService.ProxySettings(9000,

```

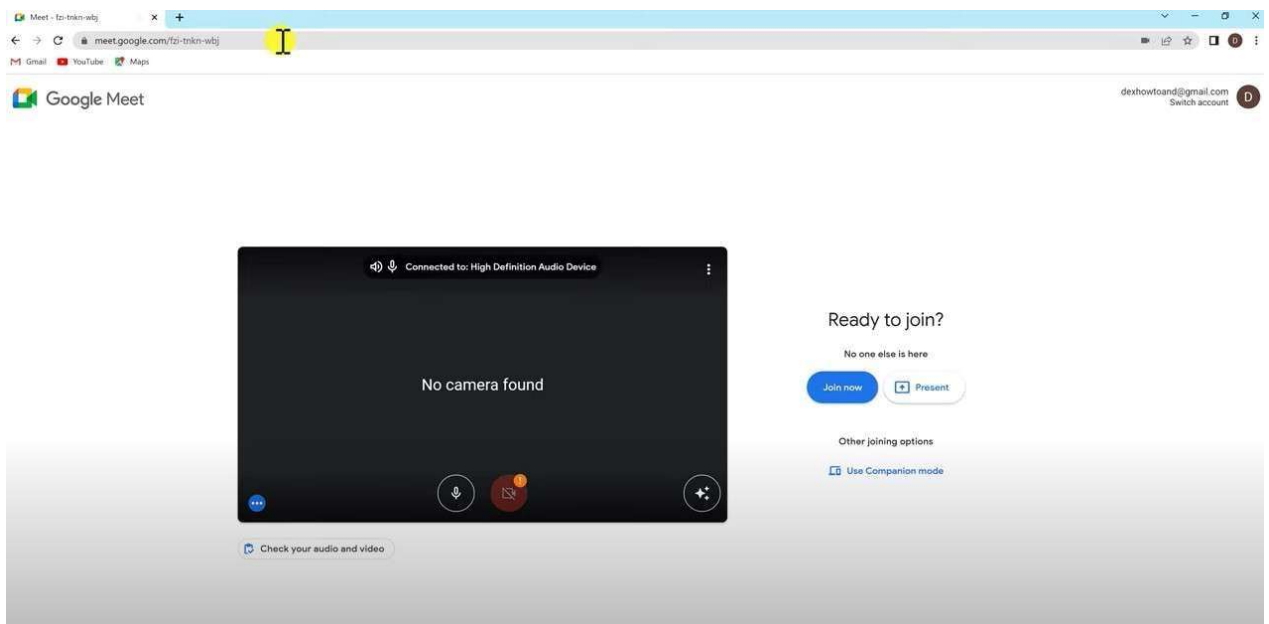
```

1      Paths.get("com/honeyfy/connectors/GoToMeetingWebConnect
2      or/Proxy/proxy.pac.js"), httpResponseFilter)) :
3      Optional.empty( );
4      try {
5          return new
6          ChromeDriverExecutorService(chromeDriverLocation,
7          "com/honeyfy/connectors/GoToMeetingWebConnector/Chro
8          me/Profile", traceFilePath, optionalProxySettings,
9          Connector.MDC_FORWARDING_KEYS);
10     }
11     catch (WebDriverException e) {
12         throw new FallBackConnector.RetryableException(null,
13         this.getClass( ), "Failed to launch Chrome", e);
14     }
15 }

```

(See *id.*, col. 15 ll. 23-43.)

46. Such techniques for interacting over the web are particularly useful for videoconferencing systems, like Google Meet, that allow joining meetings through a web browser, as shown below:



47. The '409 patent teaches that "when the conferencing software runs within a web browser, the information [necessary for recording the conference] can be gathered by scanning its document object model (DOM) and identifying the elements that contain the relevant information."

(See *id.* col. 11, ll. 24-29.) It further teaches that “[t]he DOM can also be manipulated to emulate user interaction with the application.” (See *id.*)

48. Gong’s innovations were widely recognized. For example, as one observer noted, Gong’s invention “allow[ed] reps to step away from traditional notetaking and focus on the meeting, helping them be more present.” (See Michael R. Levy, *Chorus and Gong Apps for Zoom*, GZ Consulting (August 2, 2021), <https://gzconsulting.org/2021/08/02/chorus-and-gong-apps-for-zoom/> [<https://perma.cc/96XD-UAZG>].)

49. The claims of the ’409 patent recite the use of these technological improvements, including the use a virtual participant which emulates human interactions in order to register with the conferencing system. For example, claim 1 of the ’409 patent reads as follows:

1. A method of conference recording, comprising the steps of:
 - identifying a plurality of virtual conferences being operated by a conferencing system connected to a communications network, the virtual conferences having human participants;
 - executing a plurality of virtual participant processes in a processor;
 - registering the virtual participant processes with the conferencing system as co-participants in the virtual conferences by emulating human interactions with a graphical user interface; and
 - recording information streams of the human participants using the virtual participant processes.

RECALL.AI’S INFRINGING TECHNOLOGY

50. Recall.ai markets itself as offering a “universal API for meeting bots.” (Recall.ai Home Page, <https://www.recall.ai/> [<https://perma.cc/989J-9M8J>].)

51. An API, or Application Programming Interface, allows developers to programmatically interact with a service.

52. Just as a human might interact with a piece of software using its *graphical user* interface (e.g., by opening the File menu and selecting Save), Recall.ai’s *application programming* interface allows a computer program to interact with and invoke Recall.ai’s software and services

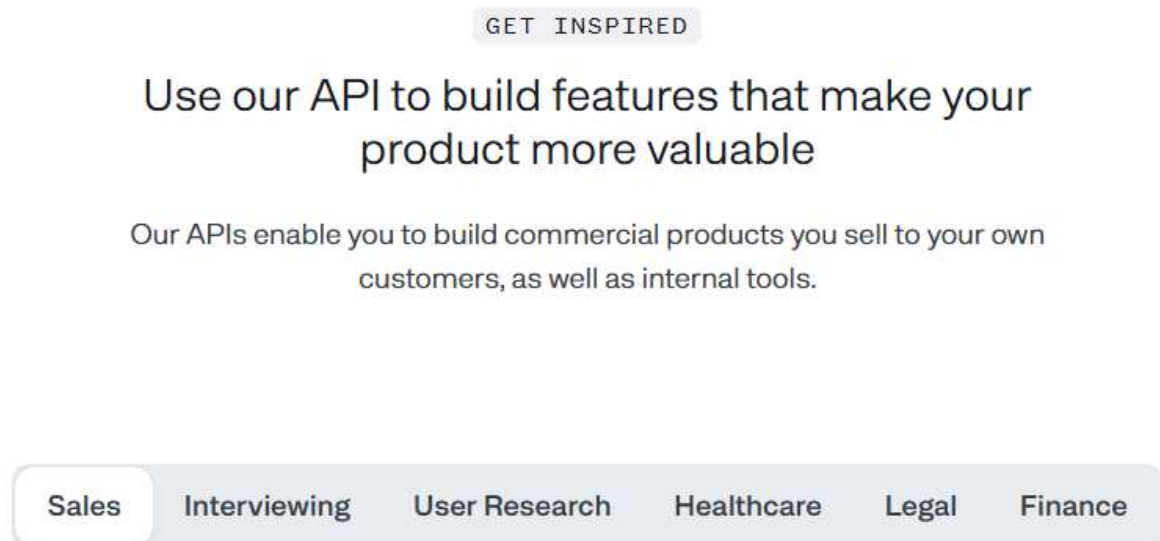
(e.g., by sending a request containing specific pieces of information over the World Wide Web to a specific Recall.ai URL in order to tell a Recall.ai bot to join a specific online meeting).

53. Just as the manual for a computer program provides users with details on its graphical user interface (e.g., what menus and submenus exist, and which options they contain), Recall.ai provides documentation to its customers to instruct them on how to use its API (e.g., which URLs they can send and receive messages from, and what information to include in requests to those URLs in order to cause Recall.ai's bots to perform certain functions or provide desired information).

54. Recall.ai offers bots that can join and record online video conferences, like Zoom meetings.

55. Using Recall.ai's API, developers can create bots, issue them commands (e.g., joining a particular meeting at a particular point in time), and obtain information from them (e.g., a recording of the meeting in the form of a video file).

56. Recall.ai's website touts its technology as allowing developers to make more valuable products in the fields of sales, interviewing, user research, healthcare, legal, and finance:



(*Id.* [<https://perma.cc/989J-9M8J>].)

57. Recall.ai's infringement of the '409 patent is knowing and willful. After being informed that it infringed the '409 patent, Recall.ai first raised a string of unmeritorious objections,

1 and then stuck its head in the sand and stopped responding to Gong altogether until this lawsuit was
2 filed.

3 58. Recall.ai has been aware of the '409 patent and its infringement no later than
4 September 20, 2024, when Gong sent Recall.ai a letter identifying the '409 patent and explaining
5 in detail how Recall.ai infringed it.

6 59. In response, Recall.ai did not substantively contest that it infringed the '409 patent.

7 60. Although Recall.ai contended that the '409 patent was invalid in light of the prior
8 art, virtually all the prior art references it identified were considered by the United States Patent
9 and Trademark Office in allowing the claims of the '409 patent to issue. The sole exception was
10 one reference that was relied upon only for the minor point that there are multiple types of video
11 conferences that a user might participate in.

12 61. On December 20, 2024, Gong sent a draft complaint to Recall.ai and followed up
13 on January 7, 2025. Recall.ai did not respond to either message.

14 62. Only after this lawsuit was filed did Recall.ai begin responding to Gong again.

15 63. Even the after-the-fact justifications that Recall.ai has developed after being sued
16 are narrow and lacking in merit.

17 64. In particular, even after this lawsuit was filed, Recall.ai has identified only one claim
18 limitation that it contends that it does not meet: "registering the virtual participant process with the
19 conferencing system as co-participants in the virtual conferences by emulating human interactions
20 with a graphical user interface."

21 65. Recall.ai's claim that it does not satisfy this limitation depends on a narrow and
22 indefensible construction of this term that would exclude several embodiments disclosed in the
23 '409 patent itself, making clear that it is merely a *post hoc* justification for Recall.ai's continued
24 infringement.

25 66. Recall.ai's continued infringement of the '409 patent with knowledge of the '409
26 patent constitutes willful infringement.

CAUSES OF ACTION**COUNT I****Infringement of U.S. Patent No. 9,699,409**

67. Gong realleges and incorporates by reference all allegations set forth in the preceding paragraphs, as if fully set forth herein.

68. Recall.ai has directly infringed and continues to infringe one or more claims of the '409 patent, literally or under the doctrine of equivalents, in violation of 35 U.S.C. § 271(a), at least because Recall.ai makes, sells, offers to sell, uses, designs, tests, and maintains within the United States products that infringe claim 1 of the '409 patent, including Recall.ai's API, bots, and related software and hardware ("Accused Product").

69. Recall.ai has contributed to the infringement of one or more claims of the '409 patent by selling, offering to sell, and/or supplying components (e.g., bot software) for use in practicing the patented methods of the '409 patent, as described below. Those components constitute a material part of the claimed invention and are not a staple article or commodity of commerce with substantial non-infringing uses, because one or more methods claimed in the '409 patent must be performed for Recall.ai's components to fulfil their intended purpose. For example, Recall.ai's bot's primary purpose is to be able to record video conferences, and doing so infringes the '409 patent.

70. Recall.ai has induced others, such as its customers, to infringe one or more of the claims of the '409 patent in violation of 35 U.S.C. § 271(b), with specific intent to induce infringement and/or willful blindness as to the possibility that its acts induce infringement. Recall.ai's customers directly infringe at least by following Recall.ai's instructions, as described in further detail below.

71. Exhibit B to this complaint sets forth a non-limiting example of Recall.ai's infringement of claim 1 of the '409 patent. A more concise explanation of Recall.ai's infringement of claim 1 of the '409 patent is set forth below.

72. To the extent the preamble of claim 1 of the '409 patent ("A method of conference recording, comprising the steps of ...") is limiting, Recall.ai instructs its customers to perform that

method. For example, Recall.ai instructs users on how “to send a bot to a meeting and retrieve the recording”:

Quickstart: Record a meeting using a bot

Learn how to send a bot to a meeting and retrieve the recording.

(Recall.ai, *Quickstart: Record a meeting using a bot*, <https://docs.recall.ai/docs/quickstart> (last visited June 24, 2025)).

4. Talk for a little bit

While you and the bot are in the meeting, make sure to turn on your video or talk for a little bit. This way, there will be actual content in the video recording the bot produces for you to look at after.

5. End the meeting

Once you feel like there is enough content in the meeting, end the meeting. The bot will automatically leave.

6. Wait for done

Once the meeting is done, Recall begins processing the video recording. This typically takes less than 10 seconds no matter the length of the meeting.

When the recording is ready to be downloaded, the bot status changes to `done`.

The best way to retrieve this is via **webhooks**. See the [Webhook Overview](#) for details.

If you can't use webhooks for some reason, you can manually **poll** [Retrieve Bot](#) to see the bot status.

Id.

7. Retrieve the recording

To retrieve the recording the bot created, use the [Retrieve Bot](#) endpoint.

- Swap the `$RECALLAI_API_KEY` placeholder in `Authorization` with your API key
- Swap the `$RECALLAI_REGION` placeholder with the region associated with your Recall account (e.g. `us-west-2`, `us-east-1`, `eu-central-1`, or `ap-northeast-1`)
- Swap the Bot ID with the id you saved in Step 3.

cURL

```
curl -X GET https://$RECALLAI_REGION.recall.ai/api/v1/bot/$BOT_ID \
-H 'Authorization: Token $RECALLAI_API_KEY'
```

The response will contain a lot of data about the meeting, but for now we'll focus on retrieving the mp4 recording. The `recordings` array in the response will contain a list of the recordings associated with this bot. In this case there's just one recording.

The mp4 recording itself is located in the `video_mixed` object in the `media_shortcuts` field of the recording object. In the `data` field, you'll see a `download_url`. Copy and paste this URL into your web browser to view.

Id.

Video and Audio

Recall bots generate a recording that can be accessed in the `media_shortcuts` field of the bot when fetched from [Retrieve Bot](#) or [List Bots](#). The field will contain a pre-signed S3 URL that you can use to display the video in your application.

For real-time applications, you can leverage websockets for accessing raw video and audio streams in [real-time](#)

(Recall.ai, *Getting Started with Recall*, <https://docs.recall.ai/docs/getting-started> (last visited June 27, 2025).)

73. Recall.ai instructs its customers to perform the step of “identifying a plurality of virtual conferences being operated by a conferencing system connected to a communications network, the virtual conferences having human participants.” For example, Recall.ai instructs its customers to synchronize a bot with their calendars so that the bot receives notifications when events, such as online meetings, are added to the calendar:

Integrate bots with your users' Google Calendar or Microsoft Outlook events.

Recall supports integrating directly with your user's calendar to simplify sending bots and syncing them with calendar events. These integrations provide a simple way to take care of event updates, bot deduplication, and other complexities that come with integrating bots with calendars.

(Recall.ai, *Getting Started*, <https://web.archive.org/web/20250427071019/https://docs.recall.ai/docs/calendar-integration>).

Once you've [successfully created a calendar in Recall](#), you can start scheduling bots to calendar events for the same using the below steps:

1. Sync Events

Once a calendar is connected, you will start receiving [calendar sync events](#) webhooks whenever an event is added/updated/removed for the calendar. For each web-hook, you should (re)fetch the calendar events via [List Calendar Events](#). You can choose to do either a full sync or use the `last_updated_ts` field in the payload (pass as `updated_at__gte` query parameter) to do an incremental sync(recommended).

Use the `is_deleted` field on the calendar event object to know if the event has been removed from the calendar or not. Recall does not delete any calendar events, and the consumer is expected to filter out events on basis of `is_deleted` attribute when syncing/displaying them to the end user.

Note that created/deleted/updated events returned by [List Calendar Events](#) will only reflect events within 1 day prior and 28 days into the future. For more info on this, see [here](#).

2. Figure out recording status of an event

After events have been synced, for each calendar event, you should decide whether it needs to be recorded. You can use the `raw` data of the calendar event in combination with your application's business logic(e.g recording preferences of a user) to decide the same. Some examples below for reference are

1. Record external events

Use the `raw` data to extract list of attendees and check those against the email of the connected calendar.

2. Record confirmed events

Use the `raw` data to extract the response of the connected calendar email

3. Record all recurring instances of an event

Store the `ical_uid` of the recurring instance as recording preference of the user. For each event compare the `ical_uid` with the stored value to decide recording status.

(Recall.ai, *Scheduling Guide*, <https://web.archive.org/web/20250324121332/https://docs.recall.ai/docs/scheduling-guide.>)

Integration Guide

The calendar integration allows every unique user in your app to connect their Google/Microsoft calendars and have Recall bots join their meetings automatically.

Integrate Recall APIs directly into your Web/Mobile applications. Refer to our [demo app](#) for an example integration ([source code here](#))

The following steps are needed for a unique user to connect their calendar and being auto-recording their meetings.

(Recall.ai, *Integration Guide*, <https://docs.recall.ai/docs/calendar-v1-integration-guide> (last visited June 24, 2025).)

74. As shown in the excerpts above, Recall.ai expects that the calendar will contain a plurality of video conferences that can be recorded. The video conferences on the calendar that is synchronized with the bot are “operated by a conferencing system connected to a communications network” (e.g., Zoom or Microsoft Teams). Finally, the video conferences each have human participants.

75. Recall.ai instructs its customers to perform the step of “executing a plurality of virtual participant processes in a processor.” For example, Recall.ai supports multiple bots, each of which is a virtual participant process in a processor:

Scheduled vs. Ad Hoc Bots

There are two different ways of using bots:

1. Ad Hoc (“On-demand”) bots - Bot joins call immediately. Very occasionally, the call to [Create Bot](#) may return a [HTTP 507](#). In this rare case, you can retry the call to Create Bot.
2. Scheduled bots - Bot is scheduled to join a call in the future, and is guaranteed to join.

We highly recommend you use scheduled bots whenever possible.

To use scheduled bots, you can either:

1. Specify the `join_at` parameter in the [Create Bot](#) endpoint at least 10 minutes in advance
2. Use our [calendar integration](#).

(Recall.ai, *Bot Fundamentals*, <https://docs.recall.ai/docs/bot-fundamentals> (last visited June 24, 2025).)

Automatically schedule bots to calendar events

The Recall calendar integration allow you to automatically schedule bots to send your users' calendar events, keeping scheduled bots in sync with any calendar changes.

For more information, check out our getting started guide [here](#).

Adding a bot email address to the meeting event

Another benefit of the Recall calendar integration is that it enables your users to add bots to meetings through inviting an email, just like you would any other participant.

To enable this flow, see [these instructions](#).

Id.

Start & Stop Recording

[Start Recording](#) triggers the bot to start recording. If a bot is already recording when this endpoint is called, a new recording will begin, overwriting the old recording.

[Stop Recording](#) stops the current recording of the bot and creates a new recording entry in the `recordings` field of the `bot`.

(Recall.ai, *Recording Control*, <https://docs.recall.ai/docs/recording-control> (last visited June 24, 2025).)

76. Recall.ai instructs its customers to perform the step of “registering the virtual participant processes with the conferencing system as co-participants in the virtual conferences by emulating human interactions with a graphical user interface.” For example, Recall.ai’s bots join video conferences.

Signed-In Google Google Meet Bots

Sign in your Google Meet bots to a Google Account

By default the Google Meet bot will join meeting as guest participant.

(Recall.ai, *Signed-In Google Google Meet Bots*, <https://docs.recall.ai/docs/google-meet-login-getting-started> (last visited June 24, 2025).)

Create your first Zoom bot

Open the Zoom **desktop** client and start a meeting.

Now that your Zoom credentials are configured in the Recall dashboard, you can send a bot to a Zoom meeting by calling [Create Bot](#).

```
cURL

curl --request POST \
  --url https://us-east-1.recall.ai/api/v1/bot/ \
  --header 'Authorization: Token {RECALL_API_KEY}' \
  --header 'accept: application/json' \
  --header 'content-type: application/json' \
  --data '{
    "meeting_url": {MEETING_URL},
    "bot_name": "My First Zoom Bot"
  }'
```

Wait a few moments and the bot will join the call.

Congrats! You just created your first Zoom bot.

(Recall.ai, *Getting Started: Zoom Bots*, <https://docs.recall.ai/docs/set-up-zoom> (last visited June 24, 2025).)²

Recording Behavior

No recording permission is needed for Microsoft Teams calls.

This means that as long as the bot is in the call, it will be able to record.

Joining Behavior

By default, Microsoft Teams bots will need to be let in from the lobby.

Authenticated Teams bots are still subject to any lobby settings enabled by the host. For instance, the host can still require that *all* participants enter through the waiting room.

(Recall.ai, *Teams Overview*, <https://docs.recall.ai/docs/microsoft-teams> (last visited June 24, 2025).)

² This screenshot shows how a programmer can instruct a Recall.ai bot to perform actions by sending a request to Recall.ai's API. By contrast, details regarding ways in which the Recall.ai bot in turn joins videoconferences are discussed in the subsequent paragraphs.

1 77. Recall.ai uses multiple techniques for registering its bots as co-participants in a
2 virtual conference “by emulating human interactions with a graphical user interface.”

3 78. For example, one technique that Recall.ai uses to register its virtual participant
4 processes with the conferencing system as co-participants in the virtual conferences by emulating
5 human interactions with a graphical user interface is to scan or manipulate a webpage’s Document
6 Object Model, or DOM.

7 79. A webpage is typically written using the Hypertext Markup Language (HTML). The
8 HTML provides the graphical user interface that a visitor to the web page sees, creating the buttons,
9 images, links, and other elements that appear on a web page.

10 80. The DOM organizes an HTML document into a hierarchical tree. For example, an
11 element corresponding to the entire body of the document may sit at the top and have “children”
12 corresponding to the title of the document and the different sections within the document (each of
13 which may in turn have further “children,” like images, paragraphs, buttons, and links).

14 81. Standard web browsers display graphical web pages by reading and interacting with
15 the elements in this DOM tree.

16 82. Other computer programs also often interact with a webpage by reading, adding,
17 removing, or modifying elements in this tree. In so doing, these programs can emulate the way in
18 which a human interacts with a web page by clicking on graphical elements present on the web
19 page.

20 83. As previously noted, scanning, and manipulating the DOM are specifically
21 mentioned in the ’409 patent as a way “to emulate user interaction with the application.” (See Ex.
22 A, ’409 patent, col. 11, *ll.* 28-29.)

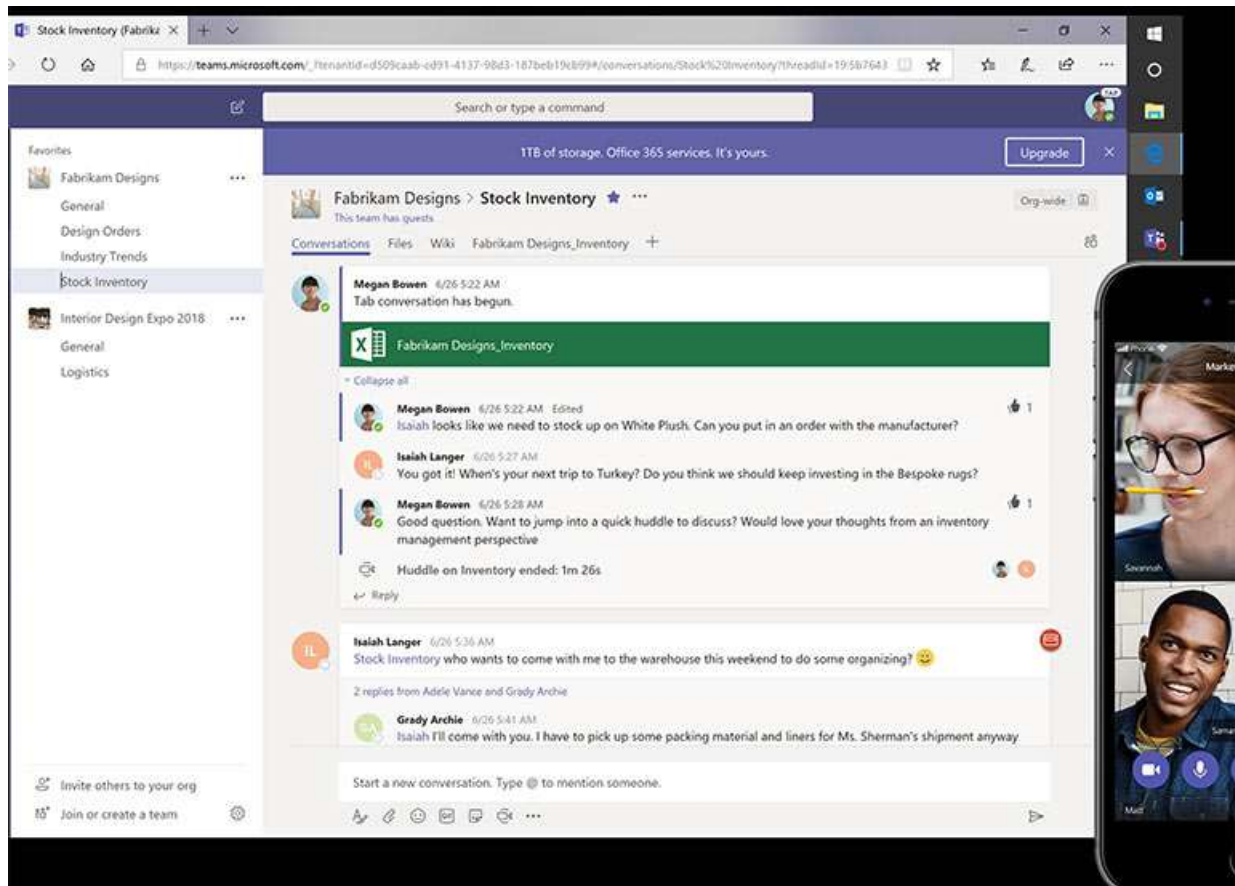
23 84. The fact that Recall.ai’s bots use the DOM or otherwise emulate a human’s
24 interactions with a graphical user interface to interface with at least some videoconferencing
25 systems is shown by the fact that Recall.ai integrates with video conferencing platforms that do not
26 provide an official API of their own. (See AiHuntList, *Recall.ai*, [https://aihuntlist.com/tool/recall-](https://aihuntlist.com/tool/recall-ai)
27 [ai](https://aihuntlist.com/tool/recall-ai) (FAQ from Recall.ai: “Can Recall.ai work without an official API for a platform? Recall.ai can
28

1 operate on any platform using just the meeting link, even if there is no official API available for
2 that platform.”); Unreal Speech, *Output Media API by Recall.ai*, [https://unrealspeech.com/ai-](https://unrealspeech.com/ai-apps/output-media-api-by-recall-ai)
3 [apps/output-media-api-by-recall-ai](https://unrealspeech.com/ai-apps/output-media-api-by-recall-ai) (“Recall.ai can operate even on platforms without an official
4 API, requiring only the meeting link to function.”).)

5 85. For example, certain videoconferencing systems, like Google Meet and Microsoft
6 Teams, do not have their own APIs. As Recall.ai notes, “Google Meet does not have an API, and
7 also can only live stream to either a special Google Meet application or YouTube. Microsoft Teams
8 does offer RTMP streaming, however has no API to configure this.” (David Gu, *How to live stream*
9 *the video from Zoom, Microsoft Teams, and Google Meet video conferences*, Recall.ai (Jan. 4,
10 2023), [https://web.archive.org/web/20230922040627/https://www.recall.ai/post/how-to-live-](https://web.archive.org/web/20230922040627/https://www.recall.ai/post/how-to-live-stream-the-video-from-zoom-microsoft-teams-and-google-meet-video-conferences)
11 [stream-the-video-from-zoom-microsoft-teams-and-google-meet-video-conferences](https://web.archive.org/web/20230922040627/https://www.recall.ai/post/how-to-live-stream-the-video-from-zoom-microsoft-teams-and-google-meet-video-conferences).)

12 86. As a result, Recall.ai’s documentation recommended using its “MS Teams Web
13 bot” to join Microsoft Teams meetings. (See Recall.ai, Teams Overview,
14 <https://web.archive.org/web/20240619080022/https://docs.recall.ai/docs/microsoft-teams>.)

15 87. A web bot takes advantage of the fact that certain videoconferencing systems allow
16 access to their platform using a graphical user interface accessible to a web browser (as shown
17 below for Microsoft Teams, which is running in a standard web browser) as opposed to using a
18 native piece of software (e.g., running the “Microsoft Teams” program on the user’s computer).
19 These web bots use the DOM of the web interface provided by the video conferencing service
20 similar to the way a standard web browser does.
21
22
23
24
25
26
27
28



88. Recall.ai also explains that in order to create a Google Meet bot, one would have to “[l]aunch an instance of the Google Meet client in a web browser.” (Amanda Zhu, *How to Integrate with Google Meet*, Recall.ai (Dec. 5, 2022), <https://web.archive.org/web/20230922050012/https://www.recall.ai/post/how-to-integrate-with-google-meet>.)

89. This technique, too, is an example of registering a bot as a co-participant in a virtual conference “by emulating human interactions with a graphical user interface.” For example, the ’409 patent discloses that this claim element can be performed by “controlling a web browser for use by a virtual participant” and provides example code for doing so. (Ex. A, ’409 patent, col. 12, ll. 31-32; *id.* col. 15, ll. 22-41.)

90. Recall.ai instructs its customers to perform the step of “recording information streams of the human participants using the virtual participant processes.” For example, Recall.ai’s bots record video conferences:

Quickstart: Record a meeting using a bot

Learn how to send a bot to a meeting and retrieve the recording.

4. Talk for a little bit

While you and the bot are in the meeting, make sure to turn on your video or talk for a little bit. This way, there will be actual content in the video recording the bot produces for you to look at after.

5. End the meeting

Once you feel like there is enough content in the meeting, end the meeting. The bot will automatically leave.

7. Retrieve the recording

To retrieve the recording the bot created, use the [Retrieve Bot](#) endpoint.

- Swap the `$RECALLAI_API_KEY` placeholder in `Authorization` with your API key
- Swap the `$RECALLAI_REGION` placeholder with the region associated with your Recall account (e.g. `us-west-2`, `us-east-1`, `eu-central-1`, or `ap-northeast-1`)
- Swap the Bot ID with the id you saved in Step 3.

cURL

```
curl -X GET https://$RECALLAI_REGION.recall.ai/api/v1/bot/$BOT_ID \
-H 'Authorization: Token $RECALLAI_API_KEY'
```

The response will contain a lot of data about the meeting, but for now we'll focus on retrieving the mp4 recording. The `recordings` array in the response will contain a list of the recordings associated with this bot. In this case there's just one recording.

The mp4 recording itself is located in the `video_mixed` object in the `media_shortcuts` field of the recording object. In the `data` field, you'll see a `download_url`. Copy and paste this URL into your web browser to view.

(Recall.ai, *Quickstart: Record a meeting using a bot*, <https://docs.recall.ai/docs/quickstart> (last visited June 24, 2025)).

91. Recall.ai's infringement has caused and continues to cause damage and irreparable harm to Gong, including loss of market share. Unless and until that infringement is enjoined by this Court, Gong will continue to suffer damage and irreparable harm as a remedy at law alone would be inadequate.

92. Recall.ai's infringement of the '409 patent is exceptional and entitles Gong to attorneys' fees and costs incurred in prosecuting this action under 35 U.S.C. § 285.

1 93. Gong is entitled to injunctive relief and damages in accordance with 35 U.S.C.
2 §§ 271, 281, 283, and 284.

3 **PRAYER FOR RELIEF**

4 94. Gong therefore respectfully requests:

- 5 a. That judgment be entered that Recall.ai has infringed at least one or more
6 claims of the '409 patent, literally and/or under the doctrine of equivalents;
7 b. An injunction enjoining Recall.ai, its officers, agents, servants, employees,
8 and attorneys, and other persons in active concert or participation with
9 Recall.ai, and its parents, subsidiaries, divisions, successors, and assigns,
10 from further infringement of the '409 patent;
11 c. An award of damages sufficient to compensate Gong for Recall.ai's
12 infringement under 35 U.S.C. § 284;
13 d. A finding that Recall.ai's infringement was willful;
14 e. That the case be found exceptional under 35 U.S.C. § 285 and that Gong be
15 awarded its reasonable attorneys' fees;
16 f. Costs and expenses in this action;
17 g. An award of pre-judgment and post-judgment interest;
18 h. That this Court, if it declines to enjoin Defendant from infringing the '409
19 patent, award damages for future infringement in lieu of an injunction; and
20 i. Such other and further relief as the Court may deem just and proper.

21 **DEMAND FOR JURY TRIAL**

22 95. Pursuant to Rule 38(b) of the Federal Rules of Civil Procedure, Gong respectfully
23 demands a trial by jury on all issues so triable.

1 Dated: June 27, 2025

Respectfully submitted,

2 ORRICK, HERRINGTON & SUTCLIFFE LLP

3 By /s/ Raghav Krishnapriyan

4 Raghav Krishnapriyan
Attorney for Plaintiff GONG.IO, INC.

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28